IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant : Peter A. Beerel, et al. Art Unit : 2133
Serial No.: 09/848,778 Examiner : Joseph Torres
Filed : May 3, 2001
Title : REDUCED-LATENCY SOFT-IN/SOFT-OUT MODULE

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

BRIEF ON APPEAL

Sir:

Applicants herewith file this brief on appeal, thereby perfecting the Notice of Appeal that was originally filed on October 27, 2006. The sections required by 37 CFR 41.37 follow.

(1) Real Party in Interest

The application is assigned to the University of Southern California, who is hence the real party in interest.

(2) Related Appeals and Interferences

There are no known related appeals and/or interferences.

(3) Status of Claims

Claims 1-19 and 36-102 are pending in the application. Each of these claims are rejected.

(4) Status of Amendments

An amendment after final was filed on September 29, 2006. An advisory action indicated that the amendment had been considered but did not place the case into condition for allowance.

(5) Summary of Claimed Subject Matter

The present application describes an improved way to decode a coded signal in a way that can increase the throughput characteristics.

Claim 1 defines a decoding method. A decoding method starts by receiving an encoded signal. Page 14 line 21 through page 15 line 18 describes the tree-based soft-in/soft-out architecture, and how it is used to find values which are used for "decoding" see page 15 line 1.

Claim 1 requires demodulating an encoded signal to produce soft information. Page 15 lines 5-11 describes that soft outputs are generated by the SISO (soft-in/soft-out) module. This is also described more specifically at page 18 lines 17 through page 21 line 9.

Claim 1 also requires iteratively processing the soft information with one or more modules that use a tree structure arranged in a parallel prefix and suffix architecture to compute

the forward and backward state metrics. Page 22 lines 4-8 describe how the operations can be carried out using tree structured architectures, and page 22 lines 11-13 describe how these are used with a combination of prefix and suffix operations. This is also more specifically described at page 23 lines 15-19. Page 24 lines 9-10 explain how this takes place sequentially in the forward and backward directions.

Claim 36 defines receiving an input signal that corresponds to one or more outputs of the finite state machine. This is described, for example, at page 9 lines 13-19 and page 18 line 8 through page 21 line 9.

Claim 36 further requires determining a soft inverse of the finite state machine by computing forward and backward metrics using a tree structure arranged in a parallel prefix and suffix architecture. See page 22 line 4 through page 23 line 12.

Claim 54 discusses a demodulator that receives a signal from a finite state machine. See, for example, page 18 line 8 through page 21 line 9. Claim 54 also defines a soft-in/soft-out module arranged in a parallel prefix and suffix architecture. See page 22 line 4 through page 23 line 12.

Claim 61 defines a method of iterative detection. This claim requires receiving an input signal corresponding to output from one or more block encoding modules. See page 18 line 8 through page 21 line 9. Claim 61 further requires then

determining the soft inverse using backward and forward state metrics using a tree structure arranged in a parallel prefix and suffix architecture. See page 22 line 4 through page 23 line 12.

Claim 78 requires a demodulator that receives an encoded signal and produces soft information indicative thereof. See page 18 line 8 through page 21 line 9. Claim 78 further requires a soft-in/soft-out module that computes a soft inverse using a tree structure arranged into parallel prefix and suffix architecture. See page 22 line 4 through page 23 line 12.

Claim 85 requires receiving an input signal whose soft inverse can be computed by running a forward-backward algorithm on a trellis representation. See page 18 line 8 through page 21 line 9. Claim 85 also requires determining the soft inverse by forwarding computing forward and backward state metrics using a tree structure arranged in parallel prefix and suffix architecture. See page 22 line 4-page 23 line 12.

Claim 91 requires receiving an encoded signal. See page 18 line 8 through page 21 line 9. Claim 91 requires demodulating it to produce soft information. See page 18 line 8 through page 21 line 9. Claim 91 further requires iteratively processing the soft information using a tree structure arranged in a parallel prefix architecture to compute forward state metrics. See page 22 line 4 through page 23 line 12.

Claim 93 defines receiving an input signal corresponding to one or more outputs of a finite state machine. See page 18 line 8 through page 21 line 9. Claim 93 requires determining a soft inverse of the finite state machine by computing forward state metrics using a tree structure arranged in a parallel prefix architecture. See page 22 line 4 through page 23 line 12.

Claim 95 defines a turbo decoder formed from a demodulator, see page 18 line 8 through page 21 line 9, see also page 9 lines 13-18 which explains that the device may be a turbo decoder. Claim 95 further requires a soft-in/soft-out module that computes a soft inverse of the finite state machine using a tree structure in a parallel prefix architecture. See page 22 line 4-page 23 line 12.

Claim 97 defines receiving an input signal corresponding to output from one or more block encoding modules. See page 18 line 8 through page 21 line 9. Claim 97 further requires determining the soft inverse of one or more of the block modules by computing forward state metrics using a parallel prefix architecture. See page 22 line 4-page 23 line 12.

Claim 99 defines a block decoder with a demodulator that receives a signal and produces soft information. See page 18 line 8 through page 21 line 9. Claim 99 further defines at least one soft-in/soft-out module that computes a soft inverse of the block including module using a tree structure arranged in

a parallel prefix architecture. See page 22 line 4-page 23 line 12.

Claim 101 defines a method including receiving an input signal corresponding to one or more outputs of the module whose soft inverse can be computed by running a forward-backward algorithm on a trellis representation. See page 18 line 8 through page 21 line 9. Claim 101 further requires determining the soft inverse by computing forward state metrics using a tree structure arranged in a parallel prefix architecture. See page 22 line 4-page 23 line 12.

(6) Grounds of Rejection to be Reviewed on Appeal

Claims 1, 2, 6-12, 15-17, 19, 36-38, 42-47, 50-52, 61-63, 67-71, 74-76, 85-90, 93, 94, 97, 98, 101 and 102 are rejected under 35 USC 103 as allegedly being unpatentable over Viterbi et al. in view of the Thomson Leighton reference. This appeal brief respectfully requests that this rejection be reviewed on appeal.

Claims 3-5, 13, 14, 18, 39-41, 48, 49, 53-60, 64-66, 72, 73, 77-84, 91, 92, 95, 96, 99 and 100 are rejected under 35 USC 103 over Viterbi et al. in view of the Thomson Leighton reference, and further in view of Benedetto et al. This appeal brief respectfully requests that this rejection be reviewed on appeal.

(7) Argument

1. Claims 1, 2, 6-12, 15-17, 19, 36-38, 42-47, 50-52, 61-63, 67-71, 74-76, 85-90, 93, 94, 97, 98, 101 and 102 are rejected under 35 USC 103 as allegedly being unpatentable over Viterbi et al. in view of the Thomson Leighton reference.

Prior to setting out some details of the argument, some background many help to frame the scope and contents of the prior art. First, note that finite state machines are often used to encode and decode very complex codes such as turbo codes. See, generally, page 15 of the specification. These codes are extremely efficient, that is they are very close to the Shannon limit for any channel. However, the codes are very complex, and may be difficult to decode. The entire system is limited by the *throughput*, the number of bits per second that the architecture, and the *latency*, end to end delay for decoding of bits. Different hardware solutions can be used to increase throughput and decrease latency.

It goes without saying that it is desirable to get the best performance possible out of any given hardware.

The bottom of page 16 describes how one standard soft-in/soft-out algorithm is the forward-backward algorithm which determines forward and backward recursion steps in parallel for

each of the finite state machines at a given time. The problem, as explained page 17 line 2, is that the architecture has a complexity that scales linearly with the block size *N*. This means that as the block size gets larger, the complexity and latency correspondingly increases. This is referred to in the specification as a linear scale solution. It leads to a natural limit on the block size that can be used for a given set of hardware.

Page 17 lines 1-9 explain, however, that using a combination of prefix and suffix operations provides an architecture that has an exponential decrease in latency for a given set.

The remainder of the specification describes the modules, and a special system that is used to improve the operation of the modules. The parallel prefix and suffix operations are described on pages 21-22. By reformulating the computation using these prefix and suffix operations, a logarithmic decrease in the latency is found. This is quite unexpected based on anything suggested by the prior art. As explained herein, there is absolutely nothing in the prior art that suggests even making the combination suggested by the rejection. Moreover, even if the combination is made, there is nothing in the prior art that teaches or makes obvious the subject matter of the claims.

The existing rejection alleges that a person having

ordinary skill in the art would make the combination of Viterbi et al. in view of the Thomson Leighton reference "to speed processing up". With all due respect, the rejection is based wholly on hindsight, and there is no motivation in the reference itself to make this kind of combination, or to make the specific combination suggested by the rejection.

First of all, Viterbi et al. teaches using serial processing to compute the state metrics. See, for example, Viterbi et al. figure 7 and figure 9, as well as the disclosure in Viterbi et al. at column 10 lines 5-39, lines 49-65, and column 11 lines 2-14. Viterbi uses a processor that serially moves forward and/or backward along the trellis and calculates the state metrics at each step. There may be multiple processors, for example the second processor moves backwards along the trellis and serially calculates the backward state metrics.

Viterbi et al. does it this way, apparently, for memory reduction, see column 9 lines 59-60, and also see column 11 lines 16-19. The purpose of Viterbi et al.'s use of serial processing is to reduce memory consumption and reduce the amount of data that needs to be stored.

Therefore, Viterbi et al.'s express teaching is to carry out a serial computation in order to reduce memory consumption. Viterbi et al.'s serial operation has the effect of reducing the

necessary amount of memory.

If Viterbi et al. were modified to use parallel computation, it would expressly contradict the teaching of Viterbi et al. and specifically the teaching of reducing memory consumption. If Viterbi et al. were modified to use parallel processing, it would no longer reduce the memory consumption in the way suggested by Viterbi et al. That is, it would contradict the express teaching of Viterbi et al, and would require modifying Viterbi et al. in a way that operated differently than the way that Viterbi et al. was intended to operate. This is exactly the kind of modification that is prohibited by the holding in *In re Gordon*, 733 F.2d 900 (Fed Cir 1984). In *Gordon*, the patent office attempted to modify a piece of prior art so that it operated in a completely different way than the way in which *Gordon* was intended to operate. The federal circuit held that modification in a way that changed the express teaching in the prior art is not proper. It is not proper to contradict the teaching of one reference in order to modify it according to the teaching of some other reference.

Viterbi et al. teaches only the use of serial suffix operations. Viterbi et al.'s column 6 lines 30-55 explains this, and the recursions in equations 5 and 6 support this. Certainly there are other ways of doing this, as suggested by the rejection made by the patent office. However, doing these

operations in this other way requires contradicting Viterbi et al.'s express teaching. It requires destroying Viterbi et al.'s goal, at least part of which is to obtain memory reduction, and to avoid storing some of these metrics. The only way to make the combination here is to take the teaching of Viterbi et al., and turn it on its head. The idea of turning teaching on its head is exactly what is prohibited by In re Gordon cited above. The hypothetical combination of Viterbi et al. in view of the Thomson Leighton reference would be an improper combination, since it would require contradicting the teaching of Viterbi et al.

The statement made in the rejection that there are many other ways of doing this is not motivation to make the combination of this reference with the Thomson Leighton reference.

Moreover, even assuming that the combination would not require turning the teaching on its head, there would still be no motivation for one having ordinary skill in the art to make this combination. Motivation, either explicitly or implicitly, must be found to make a combination of references. Viterbi et al. teaches a system that carries out serial operations and intends to reduce memory consumption. Thomson Leighton is a mathematical textbook that shows different kinds of arrays and trees. The arrays and trees can be considered for use in

parallel prefix operations, as described on pages 39-42. Note, however, the specific discussion in the Thomson Leighton reference on page 42, of the parallel prefix operation, and how it may not improve the number of steps of the algorithm at all. In fact, towards the bottom of page 42, there is an explanation that the prefix operation may take O(D) step, that is linear. Therefore, it is wholly based on hindsight to even think the Thomson Leighton reference would be combined with Viterbi et al. There is no suggestion of making the combination, and a person having ordinary skill in the art reading both references, even if they could find them, and even assuming that they could combine them based on the discussion of the above, would conclude that there would be no improvement from doing so. The Thomson Leighton reference says not one word about specific soft input/output decoders, or that its teachings could be used in such a decoder, or even that the parallel prefix operation would speed up the computation. A person having ordinary skill in the art would not be motivated to combine the Thomson Leighton reference's parallel prefix operation in place of the serial prefix operation in Viterbi et al..

In addition, applicants would like to take issue with the statement in the Official Action that "there is a multitude of prior art teaching various algorithms for using the prefix and suffix computations". Prior art that is not cited in the case,

has not been applied, and may not even have an appropriate date against the present application is quite plainly irrelevant to this rejection. To the extent the comment referred to the Thomson Leighton book pages, they are discussed herein. To the effect that this refers to some hypothetical prior art, it clearly does not meet the patent office's burden of providing a prima facie showing of unpatentability.

To summarize the above, Viterbi et al. teaches the use of serial computation. Viterbi et al. teaches that these would work perfectly well and should be used for the desired operation and would reduce memory consumption. In fact, the only teaching of using something other than serial computation for this purpose comes from the present application: not from Viterbi et al. and not from Thomson Leighton.

The rejection states that a person having ordinary skill in the art would recognize that the serial calculation is a source of the computational intensity. Again, that recognition comes from the present application. This recognition is not supported by any of the prior art in the case. In fact, on the date of the filing of the present application, there is no evidence that anyone recognized that a serial prefix computation would be computationally excessive in any way. Therefore, the contention that a very specific portion of Viterbi et al. should be replaced by the teaching in the Thomson Leighton reference is

quite clearly based on hindsight.

Moreover, even if these references were combined, there would be no teaching about using parallel prefix computation for state metrics. This is evident within Viterbi et al. itself. Viterbi et al. teaches a data dependency that inhibits the parallelization of components in order to reduce memory. That is, since there is a data dependency in Viterbi et al., it would be contrary to Viterbi et al.'s teaching to parallelize it at all—even assuming that there was incentive to do so.

In response to arguments on page 2, the rejection takes issue with our previous statement that parallel prefix computations were not taught to compute state metrics. The rejections state that Thomson Leighton teaches the use of parallel prefix computations on page 37. At the top of page 3, the official action states that one having ordinary skill in the art "only has to recognize" that forward state metrics are prefix operations. With all due respect, this recognition, made by the Official Action, is made only with the benefit of the present specification. This is much more than simply a recognition, since it requires a complete contradiction of Viterbi et al.'s use of serial processing to compute state metrics. As part of this processing, a decoder forms branch (or path) metrics and passes metrics to a processor that serially moves forward along the trellis and calculates state metrics at

every step. A second processor is given the same branch metrics and moves backwards along the trellis, again one step at a time, and serially calculates the backward state metrics along the way. A third processor then receives both forward and backwards state metrics as well as the branch metrics and creates soft outputs. Therefore the latency of the unit is O(N), where N is the length of the trellis being decoded and one section of the trellis is decoded per operation. As part of these computations for state metrics, intermediate last-in-first-out memory elements are necessary to align the data. Consequently, each state metric calculation requires that the previous state metrics have been calculated already. Therefore, Viterbi et al.'s data dependency prevents parallelization, or the application of similar hardware principles such as pipelining.

The data dependency would negate any broad principles that might influence one of ordinary skill in the art to combine the references. Given that the test of motivation to combine requires that the "combined teachings, knowledge of one of ordinary skill in the art, and the nature of the problem to be solved as a whole" is to be taken into account, a broad principle that is alleged to be implied from one reference should not be sufficient evidence of motivation to combine in view of the above, and, thus one of ordinary skill in the art would not have been motivated to combine the references. See

generally, *In re Kotzab*, 217 F.3d 1365, 1370, 55 USPQ2d 1313, 1317 (Fed. Cir. 2000).

The rejection further alleges that page 38 of the Thomson Leighton reference teaches how to overcome data dependencies. While applicants do not believe this is true, even if true, it makes applicants' own point. Specifically, if there are data dependencies, then an operation cannot be parallelized. Viterbi et al. would have to contradict his express teachings in order to parallelize the operation. This is, in effect, turning that prior art on its head.

With all due respect, therefore, the combination of these two references is in error. Moreover, even if the combination were made, a person having ordinary skill in the art would require undue experimentation to carry out parallel prefix and suffix architecture to compute the forward and backward state metrics.

Note that the rejection is effectively tantamount to the patent office saying that parallel prefix operations existed, and therefore the fact that they existed means that they could have been used in a system such as Viterbi et al. Applicants never contended that applicants invented all parallel prefix operations. In fact, these were known mathematical functions as evidenced by the Thomson Leighton reference. No one ever thought to use them in a system of the type claimed, and no one

ever suggested there would be any advantages in doing so. In fact, the Thomson Leighton reference appears to suggest that there would not be any advantages in doing so. Hence, this rejection is classic hindsight, and is not based on the teaching of the prior art.

Therefore, for all these reasons, the rejected claims should be allowable.

Claim 1 requires receding a signal, decoding it to produce soft information and iteratively processing it using a tree structure arranged in a parallel prefix and suffix architecture to compute forward and backward word state metrics. As described above, there is no teaching or suggestion of this feature in any fair combination of the prior art. Viterbi et al. only teaches serial prefix operations, and could not be modified to use parallel prefix without contradicting the express teaching. Even if it could be so modified, there is no motivation in the cited prior art to make this hypothetical combination. Even if the hypothetical combination could be made, there were just be a bare teaching of serial prefix operations from Viterbi et al. along with parallel prefix operations in general, and no teaching of how to apply the parallel prefix operations from the Thomson Leighton reference into a decoder of the type in Viterbi et al. Therefore, the entire rejection is based on hindsight, and Claim 1 should hence

be allowable.

The dependent claims which depend from Claim 1 should be allowable for similar reasons to those discussed above with respect to Claim 1.

Claim 36 further defines determining a soft inverse of the finite state machine, using state metrics using a tree structure, arranged in a parallel prefix and suffix architecture. Again, this is not taught or suggested by the cited prior art and Claim 36 should be patentable over the cited prior art, along with the claims that depend therefrom.

Claim 54 similarly defines using a tree structure arranged a parallel prefix and suffix architecture to compute a soft inverse of the finite state machine. Claim 54 was rejected over Viterbi et al. in view of the Thomson Leighton reference, and further in view of Benedetto et al. For reasons discussed above, the hypothetical combination of prior art does not teach or suggest this feature. The reference to Benedetto et al. adds the teaching of soft output decoding algorithms for turbo codes, but teaches nothing about calculating "using a tree structure arranged in a parallel prefix and suffix architecture" as claimed.

Claim 61 further defines determining the soft inverse "using a tree structure arranged in a parallel prefix and suffix architecture" something that is in no way taught or suggested by

the cited prior art.

Claim 78 defines an SISO module using a tree structure arranged in a parallel prefix and suffix architecture, something that is again not taught or suggested by the cited prior art.

Claim 85 defines determining a soft inverse using a tree structure arranged in a parallel prefix and suffix architecture. This is not taught or suggested by the hypothetical combination of prior art.

Claim 91 defines iteratively processing using a tree structure arranged in a parallel prefix architecture to compute forward state metrics. See the arguments above, this is not taught or suggested by Viterbi et al. in view of the Thomson Leighton reference, in view of Benedetto et al.

Claim 93 defines determining a soft inverse of the finite state machine by computing forward state metrics of the received input signal using a tree structure arranged in a parallel prefix architecture. This is not taught or suggested by the cited prior art.

Claim 95 defines a module that computes a soft inverse of the finite state machine using a tree structure. This is not taught or suggested by the cited prior art.

Claim 97 defines determining a soft inverse using a tree structure arranged in a parallel prefix architecture. Again, this is not taught or suggested by the cited prior art.

Claim 99 defines a soft-in/soft-out module that computes a soft inverse of the block including module using a tree structure arranged in a parallel prefix architecture.

Claim 101 defines receiving an input signal and determining its soft inverse using a tree structure arranged in a parallel prefix architecture.

All of these claims are completely patentable over the cited prior art for reasons discussed above.

2. Claims 3-5, 13, 14, 18, 39-41, 48, 49, 53-60, 64-66, 72, 73, 77-84, 91, 92, 95, 96, 99 and 100 are rejected over Viterbi et al. in view of the Thomson Leighton reference, and further in view of Benedetto et al.

The reasons why these references would not be combined have been extensively discussed above. Each of these claims are patentable over the hypothetical combination of prior art for various reasons, including that the references could not be properly combined in the way suggested by the rejection, and also because, even if combined, they do not teach or suggest the detailed features now claimed. The Benedetto et al. reference teaches various aspects of iterative decoding, but teaches nothing about the specific features now claimed.

For all of these reasons, the Examiner's rejection is in error, and should be reversed.

Please apply the appeal brief fee in the amount of $250, and any other applicable charges or credits. to Deposit Account No. 06-1050.

Respectfully submitted,

Date: December 27, 2006

Scott C. Harris
Reg. No. 32,030

Fish & Richardson P.C.
PTO Customer No. 20985
12390 El Camino Real
San Diego, California 92130
Telephone: (858) 678-5070
Facsimile: (858) 678-5099

10694906.doc

Appendix of Claims

1. A decoding method comprising:

receiving an encoded signal;

demodulating the received encoded signal to produce soft information; and

iteratively processing the soft information with one or more soft-in / soft-output (SISO) modules, at least one SISO module using a tree structure arranged to perform parallel prefix and suffix operations to compute forward and backward state metrics.

2. The method of claim 1 wherein the at least one SISO computes the forward and backward state metrics by performing recursive marginalization-combining operations.

3. The method of claim 2 wherein the recursive marginalization-combining operations comprise min-sum operations.

4. The method of claim 2 wherein the recursive marginalization-combining operations comprise min*-sum operations.

5. The method of claim 4 wherein $\min^* = \min(x,y) - \ln(1 + e^{-|x-y|})$.

6. The method of claim 2 wherein the recursive marginalization-combining operations comprise sum-product operations.

7. The method of claim 2 wherein the recursive marginalization-combining operations comprise max-product operations.

8. The method of claim 1 wherein the encoded signal comprises at least one of a turbo encoded signal, a block turbo encoded signal, a low density parity check coded signal, a product coded signal, and a convolutional coded signal.

9. The method of claim 1 wherein the encoded signal comprises at least one of a parallel concatenated convolutional code and a serial concatenated convolutional code.

10. The method of claim 1 further comprising using the iterative decoding method in a wireless communications system.

11. (The method of claim 1 further comprising terminating the iterative processing upon occurrence of a predetermined condition.

12. The method of claim 1 wherein the iterative processing comprises performing parallel prefix operations or parallel suffix operations, or both, on the soft information.

13. The method of claim 1 wherein the iterative processing comprises using soft output of a first SISO as soft input to another SISO.

14. The method of claim 1 wherein the tree structure used by at least one SISO comprises a tree structure that results in the SISO having a latency of $O(\log_2 N)$, where N is a block size.

15. The method of claim 1 wherein the tree structure used by at least one SISO comprises a Brent-Kung tree.

16. The method of claim 1 wherein the tree structure used by at least one SISO comprises a forward-backward tree.

17. The method of claim 16 wherein the forward-backward tree comprises a tree structure recursion that is bi-directional.

18. The method of claim 1 wherein the iterative processing performed by at least one SISO comprises:

tiling an observation interval into subintervals; and

applying a minimum half-window SISO operation on each subinterval.

19. The method of claim 1 wherein the iterative processing comprises performing marginalization-combining operations which form a semi-ring over the soft-information.

20-35. (Canceled).

36. A method of iterative detection comprising:

receiving an input signal corresponding to one or more outputs of a finite state machine (FSM); and

determining a soft inverse of the FSM by computing forward and backward state metrics of the received input signal using a tree structure arranged to perform parallel prefix and suffix operations.

37. The method of claim 36 wherein the forward and backward state metrics are computed by at least one soft-in / soft-out (SISO) module.

38. The method of claim 36 wherein the forward and backward state metrics are computed using a tree-structured set of marginalization-combining operations.

39. The method of claim 38 wherein the marginalization-combining operations comprise min-sum operations.

40. The method of claim 38 wherein the marginalization-combining operations comprise min*-sum operations.

41. The method of claim 40 wherein $\min^* = \min(x,y) - \ln(1 + e^{-|x-y|})$.

42. The method of claim 38 wherein the marginalization-combining operations comprise sum-product operations.

43. The method of claim 38 wherein the marginalization-combining operations comprise max-product operations.

44. The method of claim 36 wherein the input signal comprises at least one of a turbo encoded signal and a convolutional coded signal.

45. The method of claim 36 wherein the input signal comprises at least one of a parallel concatenated convolutional encoded signal and a serial concatenated convolutional encoded signal.

46. The method of claim 36 wherein determining the soft inverse of the FSM comprises iteratively processing soft information.

47. The method of claim 46 wherein the iterative processing comprises performing parallel prefix operations or parallel suffix operations, or both, on the soft information.

48. The method of claim 46 wherein the iterative processing comprises using soft output of a first SISO as soft input to another SISO.

49. The method of claim 37 wherein the tree structure used comprises a tree structure that results in the SISO module having a latency of $O(\log_2 N)$, where N is a block size.

50. The method of claim 36 wherein the tree structure comprises a Brent-Kung tree.

51. The method of claim 36 wherein the tree structure comprises a forward-backward tree.

52. The method of claim 51 wherein the forward-backward tree comprises a tree structure recursion that is bi-directional.

53. The method of claim 37 wherein the at least one SISO further:

tiles an observation interval into subintervals; and

applies a minimum half-window SISO operation on each subinterval.

54. A turbo decoder comprising:

a demodulator adapted to receive as input a signal encoded by a finite state machine (FSM) and to produce soft information relating to the received signal; and

at least one soft-in / soft-out (SISO) module in communication with the demodulator and adapted to compute a soft-inverse of the FSM using a tree structure arranged to perform parallel prefix and suffix operations.

55. The decoder of claim 54 wherein the tree structure implements a combination of parallel prefix and parallel suffix operations.

56. The decoder of claim 54 further comprising at least two SISO modules in communication with each other, wherein the SISO modules iteratively exchange soft information estimates of a decoded signal.

57. The decoder of claim 54 wherein the at least one SISO computes the soft-inverse of the FSM by computing forward and backward state metrics of the received signal.

58. The decoder of claim 54 wherein the tree structure used by the at least one SISO comprises a tree structure that results in the SISO having a latency of $O(\log_2 N)$, where N is a block size.

59. The decoder of claim 54 wherein the tree structure used by the at least one SISO comprises a Brent-Kung tree.

60. The decoder of claim 54 wherein the tree structure used by the at least one SISO comprises a forward-backward tree (FBT).

61. A method of iterative detection comprising:

receiving an input signal corresponding to output from one or more block encoding modules; and

determining the soft inverse of the one or more block encoding modules by computing forward and backward state metrics of the received input signal using a tree structure arranged to perform parallel prefix and suffix operations.

62. The method of claim 61 wherein the forward and backward state metrics are computed by at least one soft-in / soft-out (SISO) module.

63. The method of claim 61 wherein the forward and backward state metrics are computed using a tree-structured set of marginalization-combining operations.

64. The method of claim 63 wherein the marginalization-combining operations comprise min-sum operations.

65. The method of claim 63 wherein the marginalization-combining operations comprise min*-sum operations.

66. (The method of claim 65 wherein $\min^* = \min(x,y) - \ln(1 + e^{-|x-y|})$.

67. The method of claim 63 wherein the marginalization-combining operations comprise sum-product operations.

68. The method of claim 63 wherein the marginalization-combining operations comprise max-product operations.

69. The method of claim 63 wherein the input signal comprises at least one of a block turbo encoded signal, a low density parity check coded signal, and a product coded signal.

70. The method of claim 63 wherein determining the soft inverse of the one or more block encoding modules comprises iteratively processing soft information.

71. The method of claim 70 wherein the iterative processing comprises performing parallel prefix operations or parallel suffix operations, or both, on the soft information.

72. The method of claim 70 wherein the iterative processing comprises using soft output of a first SISO as soft input to another SISO.

73. The method of claim 62 wherein the tree structure used comprises a tree structure that results in the SISO module having a latency of O(log2 N), where N is a block size.

74. The method of claim 61 wherein the tree structure comprises a Brent-Kung tree.

75. The method of claim 61 wherein the tree structure comprises a forward-backward tree.

76. The method of claim 75 wherein the forward-backward tree comprises a tree structure recursion that is bi-directional.

77. The method of claim 62 wherein the at least one SISO further:

tiles an observation interval into subintervals; and

applies a minimum half-window SISO operation on each subinterval.

78. A block decoder comprising:

a demodulator adapted to receive as input a signal encoded by a block encoding module and to produce soft information relating to the received signal; and

at least one soft-in / soft-out (SISO) module in communication with the demodulator and adapted to compute a soft-inverse of the block encoding module using a tree structure arranged to perform parallel prefix and suffix operations.

79. The decoder of claim 78 wherein the tree structure implements a combination of parallel prefix and parallel suffix operations.

80. The decoder of claim 78 further comprising at least two SISO modules in communication with each other, wherein the SISO modules iteratively exchange soft information estimates of a decoded signal.

81. The decoder of claim 78 wherein at least one SISO computes the soft-inverse of the block encoding module by computing forward and backward state metrics of the received signal.

82. The decoder of claim 78 wherein the tree structure used by at least one SISO comprises a tree structure that results in the SISO having a latency of O(log2 N), where N is a block size.

83. The decoder of claim 78 wherein the tree structure used by at least one SISO comprises a Brent-Kung tree.

84. The decoder of claim 78 wherein the tree structure used by at least one SISO comprises a forward-backward tree (FBT).

85. An iterative detection method comprising:

receiving an input signal corresponding to one or more outputs of a module whose soft-inverse can be computed by running a forward-backward algorithm on a trellis representation of the module; and

determining the soft inverse of the module by computing forward and backward state metrics of the received input signal

using a tree structure arranged to perform parallel prefix and suffix operations.

86. The method of claim 85 wherein the input signal comprises at least one of a block error correction encoded signal, a block turbo encoded signal, a low density parity check coded signal, and a product coded signal.

87. The method of claim 85 wherein the input signal comprises at least one of a turbo encoded signal and a convolutional coded signal.

88. The method of claim 85 wherein the encoded signal comprises at least one of a parallel concatenated convolutional code and a serial concatenated convolutional code.

89. The method of claim 85 wherein the module comprises a finite state machine.

90. The method of claim 85 wherein the module comprises a block encoding module.

91. A decoding method comprising:

receiving an encoded signal;

demodulating the received encoded signal to produce soft information; and

iteratively processing the soft information with one or more soft-in / soft-output (SISO) modules, at least one SISO module using a tree structure to perform parallel prefix operations to compute forward state metrics.

92. The method of claim 91 wherein the tree structure is further arranged to perform parallel suffix operations to compute backward state metrics.

93. A method of iterative detection comprising:

receiving an input signal corresponding to one or more outputs of a finite state machine (FSM); and

determining a soft inverse of the FSM by computing forward state metrics of the received input signal using a tree structure arranged to perform parallel prefix operations.

94. The method of claim 93 wherein the tree structure is further arranged to perform parallel suffix operations and determining the soft inverse of the FSM further comprises determining the soft inverse of the FSM by computing backward state metrics of the received input signal using the tree structure.

95. A turbo decoder comprising:

a demodulator adapted to receive as input a signal encoded by a finite state machine (FSM) and to produce soft information relating to the received signal; and

at least one soft-in / soft-out (SISO) module in communication with the demodulator and adapted to compute a soft-inverse of the FSM using a tree structure arranged to perform parallel prefix operations.

96. The decoder of claim 95 wherein the tree structure is further arranged to perform parallel suffix operations.

97. A method of iterative detection comprising:

receiving an input signal corresponding to output from one or more block encoding modules; and

determining the soft inverse of the one or more block encoding modules by computing forward state metrics of the received input signal using a tree structure arranged to perform parallel prefix operations.

98. The method of claim 97 wherein the tree structure is further arranged to perform parallel suffix operations and determining the soft inverse of the one or more block encoding

modules further comprises determining the soft inverse of the one or more block encoding modules by computing backward state metrics of the received input signal using the tree structure.

99. A block decoder comprising:

a demodulator adapted to receive as input a signal encoded by a block encoding module and to produce soft information relating to the received signal; and

at least one soft-in / soft-out (SISO) module in communication with the demodulator and adapted to compute a soft-inverse of the block encoding module using a tree structure to perform parallel prefix operations.

100. The decoder of claim 99 wherein the tree structure is further arranged to perform parallel suffix operations.

101. An iterative detection method comprising:

receiving an input signal corresponding to one or more outputs of a module whose soft-inverse can be computed by running a forward-backward algorithm on a trellis representation of the module; and

determining the soft inverse of the module by computing forward state metrics of the received input signal using a tree structure arranged to perform parallel prefix operations.

102. The method of claim 101 wherein the tree structure is further arranged to perform parallel suffix operations and determining the soft inverse of the module further comprises determining the soft inverse of the module by computing backward state metrics of the received input signal using the tree structure.

Evidence Appendix

None.

Related Proceedings Appendix

None.